

EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT

A large grid of 1000 characters (SS, YY, VV, AA, XX) arranged in a pattern that resembles a stylized '1000'. The characters are arranged in a grid where the first column contains 10 'SS's, the second column contains 10 'YY's, the third column contains 10 'YY's, the fourth column contains 10 'SS's, the fifth column contains 10 'VV's, the sixth column contains 10 'VV's, the seventh column contains 10 'AA's, the eighth column contains 10 'AA's, the ninth column contains 10 'XX's, and the tenth column contains 10 'XX's. The pattern is a stylized '1000'.

```

LL          IIIII
LL          IIIII
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LLLLLLLLLLL IIIII
LLLLLLLLLLL IIIII

                SSSSSSSS
                SSSSSSSS
                SS
                SS
                SS
                SS
                SSSSSS
                SSSSSS
                                SS
                                SS
                                SS
                                SS
                SSSSSSSS
                SSSSSSSS

```

```
0001 0 XTITLE 'EDT$SYSVAX - VAX/VMS system specific storage'
0002 0 MODULE EDT$SYSVAX ( ! VAX/VMS system specific storage
0003 0 IDENT = 'V04-000' ! File: SYSVAX.B32 Edit: JBS2034
0004 0 ) =
0005 1 BEGIN
0006 1
0007 1 *****
0008 1 *
0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0011 1 * ALL RIGHTS RESERVED.
0012 1 *
0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0018 1 * TRANSFERRED.
0019 1 *
0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0022 1 * CORPORATION.
0023 1 *
0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0026 1 *
0027 1 *****
0028 1
0029 1
0030 1
0031 1 ++
0032 1 FACILITY: EDT -- The DEC Standard Editor
0033 1
0034 1 ABSTRACT:
0035 1
0036 1 This module contains system specific code for the VAX/VMS
0037 1 environment.
0038 1
0039 1 ENVIRONMENT: VAX/VMS only
0040 1
0041 1 AUTHOR: Bob Kushlis, CREATION DATE: March 22, 1979
0042 1
0043 1 MODIFIED BY:
0044 1
0045 1 Bob Kushlis, 10-JUL-1979
0046 1 Convert the case of the file names.
0047 1 John Sauter, 19-Dec-1980, 02
0048 1 Add tracing.
0049 1 2-003 - Regularize the headers. JBS 19-Feb-1981
0050 1 2-004 - Allocate an event flag for the "working" message, and make it
0051 1 cancel only its own timers. JBS 19-Feb-1981
0052 1 2-005 - Fix module header and certain symbols. JBS 30-Mar-1981
0053 1 2-006 - Stop the "working" message only if it is running. JBS 02-Apr-1981
0054 1 2-007 - Implement the virtual deallocation routine. TMV 6-Aug-81
0055 1 2-008 - EDT$SALO HEAP should return 1 if successful, 0 if not.
0056 1 JBS 07-Aug-1981
0057 1 2-009 - Remove calls to LIB$SHOW_VM. JBS 21-Aug-1981
```



```
.. 58      0058 1 2-010 - Add date/time routine. STS 02-Sep-1981
.. 59      0059 1 2-011 - Add deallocation of text string area. STS 06-Oct-1981
.. 60      0060 1 2-012 - Always do deallocation of text and entity string areas. STS 06-Nov-1981
.. 61      0061 1 2-013 - Add global for SET/SHOW HELP command. SMB 16-Dec-1981
.. 62      0062 1 2-014 - Revise timer AST logic. JBS 13-Jan-1982
.. 63      0063 1 2-015 - Change 32-bit line# to 48 bit. SMB 16-Jan-1982
.. 64      0064 1 2-016 - Move line number declarations to DATA.BLI. SMB 29-Jan-1982
.. 65      0065 1 2-017 - Take out extra space in date when day is single digit. STS 02-Feb-1982
.. 66      0066 1 2-018 - Fix a race condition in timer AST logic. JBS 10-Feb-1982
.. 67      0067 1 2-019 - Take out call to sys$exit. STS 19-Feb-1982
.. 68      0068 1 2-020 - Add edt$$z_wf_desc to deallocation list. STS 09-Mar-1982
.. 69      0069 1 2-021 - Define the default startup file names. JBS 18-Mar-1982
.. 70      0070 1 2-022 - Correct the length of EDTINI. JBS 08-Apr-1982
.. 71      0071 1 2-023 - Change the HELP file default name. SMB 10-May-1982
.. 72      0072 1 2-024 - Put the default startup file on SYS$LIBRARY. JBS 08-Jun-1982
.. 73      0073 1 2-025 - Erase the working message line in STOP WKINGMSG. SMB 28-Jun-1982
.. 74      0074 1 2-026 - New implementation of defined keys. JBS 12-Aug-1982
.. 75      0075 1 2-027 - Change the command file name. JBS 23-Aug-1982
.. 76      0076 1 2-028 - Change the command file name again. JBS 17-Sep-1982
.. 77      0077 1 2-029 - Change EDT$$FMT LIT to EDT$$FMT STR. JBS 05-Oct-1982
.. 78      0078 1 2-030 - Remove deallocation of edt$$z_wf_desc. STS 11-Nov-1982
.. 79      0079 1 2-031 - Add a hack to debug insufficient memory problems. JBS 15-Nov-1982
.. 80      0080 1 2-032 - Add a call to deassign terminal channel. STS 21-Dec-1982
.. 81      0081 1 2-033 - Deassign the terminal channel before halting trace, since the
.. 82      0082 1      terminal deassign may output a keypad setting. JBS 26-Apr-1983
.. 83      0083 1 2-034 - Improve the appearance of the listing. JBS 17-Jun-1983
.. 84      0084 1 --
.. 85      0085 1
```

```

87 0086 1 $SBTTL 'Declarations'
88 0087 1
89 0088 1 TABLE OF CONTENTS:
90 0089 1
91 0090 1
92 0091 1 REQUIRE 'EDT$SRC:TRAROUNAM';
93 0530 1
94 0531 1 FORWARD ROUTINE
95 0532 1     EDT$INTER_ERR : NOVALUE,
96 0533 1     EDT$SYS_EXI : NOVALUE,
97 0534 1     EDT$GET_DATE : NOVALUE,
98 0535 1     EDT$ALO_HEAP,
99 0536 1     EDT$DEA_HEAP : NOVALUE,
100 0537 1     EDT$DEA_ALLHEAP : NOVALUE,
101 0538 1     WORKAST : NOVALUE,
102 0539 1     EDT$START_WKINGMSG : NOVALUE,
103 0540 1     EDT$STOP_WKINGMSG : NOVALUE,
104 0541 1     EDT$MSG_TOSTR : NOVALUE;
105 0542 1
106 0543 1
107 0544 1 INCLUDE FILES:
108 0545 1
109 0546 1
110 0547 1 REQUIRE 'EDT$SRC:SYSSYM';
111 0577 1
112 0578 1 REQUIRE 'EDT$SRC:EDTREQ';
113 0713 1
114 0714 1 LIBRARY 'EDT$SRC:KEYPADDEF';
115 0715 1
116 0716 1 REQUIRE 'TRACESEL';
117 0747 1
118 0748 1 REQUIRE 'EDT$SRC:TRACEMAC';
119 0975 1
120 0976 1
121 0977 1 MACROS:
122 0978 1
123 0979 1     NONE
124 0980 1
125 0981 1 EQUATED SYMBOLS:
126 0982 1
127 0983 1     NONE
128 0984 1
129 0985 1 OWN STORAGE:
130 0986 1
131 0987 1
132 0988 1 GLOBAL
133 0989 1     EDT$ST_HDEF_NAM : BLOCK [14, BYTE] INITIAL (BYTE (13, 'SYS$HELP:.HLB')),
134 0990 1     EDT$ST_HDEF_FILE : BLOCK [8, BYTE] INITIAL (BYTE (7, 'EDTHELP')),
135 0991 1     EDT$ST_HELP_NAM : BLOCK [NAM$C MAXRSS, BYTE] INITIAL (BYTE ('EDTHELP')),
136 0992 1     EDT$G_HELP_NAMLEN : INITIAL (7),
137 0993 1     EDT$G_HELP_SET : INITIAL (0),
138 0994 1     EDT$Z_LBR_INDEX, ! LBR Control index for HELP
139 0995 1     EDT$ST_CMD_NAM_DEF1 : BLOCK [7, BYTE] INITIAL (BYTE (6, 'EDTSYS')), ! Command file name
140 0996 1     EDT$ST_CMD_NAM_DEF2 : BLOCK [17, BYTE] INITIAL (BYTE (16, 'SYS$LIBRARY:.EDT')), ! Command file default name
141 0997 1
142 0998 1     EDT$ST_CMD_NAM_DEF3 : BLOCK [7, BYTE] INITIAL (BYTE (6, 'EDTINI')), ! Alternate command file name
143 0999 1     EDT$ST_CMD_NAM_DEF4 : BLOCK [5, BYTE] INITIAL (BYTE (4, '.EDT')); ! Alternate command file default nam
```



```
144 1000 1
145 1001 1 OWN
146 1002 1 MESSAGE : VECTOR [12, BYTE] INITIAL (BYTE ('Bug check '));
147 1003 1
148 1004 1 OWN
149 1005 1 DEL TIME : VECTOR [2] INITIAL (-5000000, -1),
150 1006 1 WORKING_EFN,
151 1007 1 WORK_MESSAGE_RUNNING : VOLATILE INITIAL (0);
152 1008 1
153 1009 1 OWN
154 1010 1 MEM_USE : INITIAL (0), ! Currently allocated memory amount
155 1011 1 MEM_LIMIT : INITIAL (1000000000); ! Limit on amount of memory to allocate
156 1012 1
157 1013 1
158 1014 1 ! EXTERNAL REFERENCES:
159 1015 1 !
160 1016 1
161 1017 1 EXTERNAL ROUTINE
162 1018 1 EDT$STI_WSTR,
163 1019 1 EDT$SOUT_FMTBUF,
164 1020 1 EDT$SSC_POSCSIF,
165 1021 1 EDT$SSC_ERATOEOI,
166 1022 1 EDT$STI_WRLN : NOVALUE,
167 1023 1 EDT$FMT_STR : NOVALUE,
168 1024 1 LIB$GET_VM,
169 1025 1 LIB$FREE_VM,
170 1026 1 SYS$EXIT,
171 1027 1 LIB$DATE_TIME,
172 1028 1 LIB$GET_EF,
173 1029 1 LIB$FREE_EF;
174 1030 1
175 1031 1 !+
176 1032 1 ! Define the RABs to be used by EDT
177 1033 1 !-
178 1034 1
179 1035 1 GLOBAL
180 1036 1 EDT$SZ_SYS_PRIAB : $RAB_DECL,
181 1037 1 EDT$SZ_SYS_JOURAB : $RAB_DECL,
182 1038 1 EDT$SZ_SYS_CMDRAB : $RAB_DECL,
183 1039 1 EDT$SZ_SYS_ALTRAB : $RAB_DECL;
184 1040 1
185 1041 1 EXTERNAL
186 1042 1 EDT$SA_FMT_WRRUT, ! Output format routine
187 1043 1 EDT$SG_MESSAGE_LINE, ! Command/message line
188 1044 1 EDT$SG_SECOND : VOLATILE, ! Set to 1 once a second for WORKING message
189 1045 1 EDT$SG_WORKCOUNT; ! Counter to support WORKING message
190 1046 1
```

```
192 1047 1 %SBTTL 'EDT$$INTER_ERR - internal error'
193 1048 1
194 1049 1 GLOBAL ROUTINE EDT$$INTER_ERR ! Internal error
195 1050 1 : NOVALUE =
196 1051 1
197 1052 1 ++
198 1053 1 FUNCTIONAL DESCRIPTION:
199 1054 1
200 1055 1 If an internal error is detected in EDT, come here to
201 1056 1 print a cryptic message and bail out.
202 1057 1
203 1058 1 FORMAL PARAMETERS:
204 1059 1
205 1060 1 NONE
206 1061 1
207 1062 1 IMPLICIT INPUTS:
208 1063 1
209 1064 1 NONE
210 1065 1
211 1066 1 IMPLICIT OUTPUTS:
212 1067 1
213 1068 1 NONE
214 1069 1
215 1070 1 ROUTINE VALUE:
216 1071 1
217 1072 1 NONE
218 1073 1
219 1074 1 SIDE EFFECTS:
220 1075 1
221 1076 1 Never returns to its caller.
222 1077 1
223 1078 1 --
224 1079 1
225 1080 2 BEGIN
226 1081 2 MESSAGES ((INTERERR));
227 1082 2 SIGNAL_STOP (EDT$_INTERERR);
228 1083 1 END; ! of routine EDT$$INTER_ERR
```

```
.TITLE EDTSSYSVAX EDTSSYSVAX - VAX/VMS system specific
storage
```

```
.IDENT \V04-000\
```

```
.PSECT _EDT$DATA,NOEXE, PIC,2
```

```
0D 00000 EDT$$T_HDEF_NAM::
42 4C 48 2E 3A 50 4C 45 48 24 53 59 53 00001 .BYTE 13
0000E .ASCII \SYSS$HELP:.HLB\
07 00010 EDT$$T_HDEF_FILE::
50 4C 45 48 54 44 45 00011 .BYTE 7
50 4C 45 48 54 44 45 00018 EDT$$T_HELP_NAM::
0001F .ASCII \EDTHELP\
00117 .BLKB 248
00000007 00118 EDT$$G_HELP_NAMLEN::
```

EDT\$SYSVAX
V04-000

EDT\$SYSVAX - VAX/VMS system specific storage
EDT\$\$INTER_ERR - internal error

F 5
16-Sep-1984 01:52:10
14-Sep-1984 12:24:48

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]SYSVAX.B32;1 Page 6
(3)

```

                                .LONG 7
00000000 0011C EDT$$G_HELP SET::
                                .LONG 0
                                00120 EDT$$Z_LBR_INDEX::
                                .BLKB 4
06 00124 EDT$$T_CMD_NAM_DEF1::
                                .BYTE 6
53 59 53 54 44 45 00125 .ASCII \EDTSYS\
0012B .BLKB 1
10 0012C EDT$$T_CMD_NAM_DEF2::
                                .BYTE 16
44 45 2E 3A 59 52 41 52 42 49 4C 24 53 59 53 0012D .ASCII \SYS$LIBRARY:.EDT\
54 0013C
0013D .BLKB 3
06 00140 EDT$$T_CMD_NAM_DEF3::
                                .BYTE 6
49 4E 49 54 44 45 00141 .ASCII \EDTINI\
00147 .BLKB 1
04 00148 EDT$$T_CMD_NAM_DEF4::
                                .BYTE 4
54 44 45 2E 00149 .ASCII \.EDT\
0014D .BLKB 3
20 20 20 6B 63 65 68 63 20 67 75 42 00150 MESSAGE:.ASCII \Bug check \
FFFFFFFF FFB3B4C0 0015C DEL_TIME:
                                .LONG -5000000, -1
00164 WORKING_EFN:
                                .BLKB 4
00000000 00168 WORK_MESSAGE RUNNING:
                                .LONG 0
00000000 0016C MEM_USE:.LONG 0
3B9ACA00 00170 MEM_LIMIT:
                                .LONG 1000000000
00174 EDT$$Z_SYS_PRIAB::
                                .BLKB 68
001B8 EDT$$Z_SYS_JOURAB::
                                .BLKB 68
001FC EDT$$Z_SYS_CMDAB::
                                .BLKB 68
00240 EDT$$Z_SYS_ALTRAB::
                                .BLKB 68

                                .EXTRN EDT$$TI_WSTR, EDT$$OUT_FMTBUF
                                .EXTRN EDT$$SC_POSCSIF
                                .EXTRN EDT$$SC_ERATOEOI
                                .EXTRN EDT$$TI_WRLN, EDT$$FMT_STR
                                .EXTRN LIB$GET_VM, LIB$FREE_VM
                                .EXTRN SYS$EXIT, LIB$DATE TIME
                                .EXTRN LIB$GET_EF, LIB$FREE_EF
                                .EXTRN EDT$$A_FMT_WRRUT
                                .EXTRN EDT$$G_MESSAGE_LINE
                                .EXTRN EDT$$G_SECOND, EDT$$G_WORKCOUNT
                                .EXTRN EDT$_INTERERR

                                .PSECT _EDT$CODE, NOWRT, SHR, PIC, 2

                                .ENTRY EDT$$INTER_ERR, Save nothing
00000000G 8F DD 00002 PUSHL #EDT$_INTERERR

```

: 1049
: 1082

EDTSSYSVAX
V04-000

EDTSSYSVAX - VAX/VMS system specific storage
EDTSSINTER_ERR - internal error

^{G 5}
16-Sep-1984 01:52:10
14-Sep-1984 12:24:48

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]SYSVAX.B32;1 Page 7
(3)

00000000G 00

01 FB 00008
04 0000F

CALLS #1, LIB\$STOP
RET

: 1083

; Routine Size: 16 bytes, Routine Base: _EDT\$CODE + 0000

; 229 1084 1

```

231 1085 1 %SBTTL 'EDTSSYS_EXI - exit back to the operating system'
232 1086 1
233 1087 1 GLOBAL ROUTINE EDTSSYS_EXI (
234 1088 1     STATUS
235 1089 1     ) : NOVALUE =
236 1090 1
237 1091 1 ++
238 1092 1 FUNCTIONAL DESCRIPTION:
239 1093 1
240 1094 1     Final clean-up
241 1095 1
242 1096 1 FORMAL PARAMETERS:
243 1097 1
244 1098 1     STATUS          Exit status code. 1 = normal.
245 1099 1
246 1100 1 IMPLICIT INPUTS:
247 1101 1
248 1102 1     NONE
249 1103 1
250 1104 1 IMPLICIT OUTPUTS:
251 1105 1
252 1106 1     NONE
253 1107 1
254 1108 1 ROUTINE VALUE:
255 1109 1
256 1110 1     NONE
257 1111 1
258 1112 1 SIDE EFFECTS:
259 1113 1
260 1114 1     Deallocates all heap memory
261 1115 1
262 1116 1 --
263 1117 1
264 1118 2 BEGIN
265 1119 2
266 1120 2 EXTERNAL ROUTINE
267 1121 2     EDT$TI_DEAS;
268 1122 2
269 1123 2 MESSAGES ((EDITORABO));
270 1124 2 EDT$DEA_ALLHEAP ();
271 1125 2 EDT$TI_DEAS ();
272 1126 2
273 1127 2 %IF EDT$STR_ACT
274 1128 2 %THEN
275 1129 2 BEGIN
276 1130 2
277 1131 2 LOCAL
278 1132 2     TRACE_STATUS;
279 1133 2
280 1134 2 EXTERNAL ROUTINE
281 1135 2     EDT$STR_CLS : ADDRESSING_MODE (GENERAL);
282 1136 2
283 1137 2 EXTERNAL
284 1138 2     EDT$SL_TR_INFLG;
285 1139 2
286 1140 2 $$TRACE (EDT$STR_EXI, EDT$STR_SEXI, 0, 0);
287 1141 2 TRACE_STATUS = EDT$STR_CLS (EDT$SL_TR_INFLG);
```

EDTSSYSVAX
V04-000

EDTSSYSVAX - VAX/VMS system specific storage
EDTSSSYS_EXI - exit back to the operating syst

1 5
16-Sep-1984 01:52:10
14-Sep-1984 12:24:48

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]SYSVAX.B32;1

Page 9
(4)

```

: 288
: 289
: 290
: 291
: 292
: 293
: 294
: 295
: 296

U 1142 2
U 1143 2
U 1144 2
U 1145 2
U 1146 2
U 1147 2
U 1148 2
U 1149 2
U 1150 1

IF ( NOT .TRACE_STATUS) THEN SIGNAL_STOP (.TRACE_STATUS);
END;
%FI
IF ( NOT .STATUS) THEN SIGNAL_STOP (EDT$_EDITORABO);
END;
! of routine EDTSSSYS_EXI
```

```

0000V CF 0000 0000
00000000G 00 00 FB 00002
00000000G 00 00 FB 00007
00000000G 00 04 AC E8 0000E
00000000G 00 8F DD 00012
00000000G 00 01 FB 00018
04 0001F 1$:
```

.EXTRN EDT\$STI_DEAS, EDT\$_EDITORABO

```

.ENTRY EDTSSSYS_EXI, Save nothing
CALLS #0, EDT$DEA_ALLHEAP
CALLS #0, EDT$STI_DEAS
BLBS STATUS, 1$
PUSHL #EDT$_EDITORABO
CALLS #1, LIB$STOP
RET
```

```

: 1087
: 1124
: 1125
: 1148
: 1150
```

; Routine Size: 32 bytes. Routine Base: _EDT\$CODE + 0010

; 297 1151 1


```
299 1152 1 XSBTTL 'EDTSSGET_DATE - return the date as an ASCII string'
300 1153 1
301 1154 1 GLOBAL ROUTINE EDTSSGET_DATE (
302 1155 1     LEN
303 1156 1     BUFFER
304 1157 1 ) : NOVALUE =
305 1158 1
306 1159 1 ++
307 1160 1 FUNCTIONAL DESCRIPTION:
308 1161 1
309 1162 1     Return the date and time as an ASCII string.
310 1163 1
311 1164 1 FORMAL PARAMETERS:
312 1165 1
313 1166 1     LEN                Length of the buffer in which the date is returned
314 1167 1
315 1168 1     BUFFER             Address of that buffer.
316 1169 1
317 1170 1 IMPLICIT INPUTS:
318 1171 1
319 1172 1     NONE
320 1173 1
321 1174 1 IMPLICIT OUTPUTS:
322 1175 1
323 1176 1     NONE
324 1177 1
325 1178 1 ROUTINE VALUE:
326 1179 1
327 1180 1     NONE
328 1181 1
329 1182 1 SIDE EFFECTS:
330 1183 1
331 1184 1     NONE
332 1185 1
333 1186 1 --
334 1187 1
335 1188 2 BEGIN
336 1189 2
337 1190 2 LOCAL
338 1191 2     DATE_DESC : BLOCK [8, BYTE],
339 1192 2     DATE_TIME_STATUS;
340 1193 2
341 1194 2 MAP
342 1195 2     BUFFER : REF VECTOR [, BYTE];
343 1196 2
344 1197 2 ++
345 1198 2 Set up the descriptor for the LIB$ routine
346 1199 2 --
347 1200 2     DATE_DESC [DSC$W_LENGTH] = 24;
348 1201 2     DATE_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
349 1202 2     DATE_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
350 1203 2     DATE_DESC [DSC$A_POINTER] = BUFFER [1];
351 1204 2 ++
352 1205 2 Now call the routine to get the date and time as string
353 1206 2 --
354 1207 2     DATE_TIME_STATUS = LIB$DATE_TIME (DATE_DESC);
355 1208 2 ++
```

```

356 1209 2 ! Make sure we got a good status from the library routine else stop
357 1210 2 !
358 1211 2
359 1212 2 IF ( NOT .DATE_TIME_STATUS) THEN SIGNAL_STOP (.DATE_TIME_STATUS);
360 1213 2
361 1214 2 BUFFER [0] = %C' '; ! begin with a space
362 1215 2 BUFFER [21] = %C' i; ! and end with a space
363 1216 2
364 1217 2 IF (.BUFFER [1] EQL %C' ')
365 1218 2 THEN
366 1219 2 BEGIN
367 1220 2 CH$MOVE (20, BUFFER [2], BUFFER [1]); ! shift left one space
368 1221 2 .LEN = 21;
369 1222 2 END
370 1223 2 ELSE
371 1224 2 .LEN = 22;
372 1225 2
373 1226 1 END; ! of routine EDT$GET_DATE
```

				003C 00000	.ENTRY EDT\$GET_DATE, Save R2,R3,R4,R5	1154
	SE			04 C2 00002	SUBL2 #4, SP	
		010E0018		8F DD 00005	PUSHL #17694744	1200
	52	08		AC DD 0000B	MOVL BUFFER, R2	1203
	04 AE	01		A2 9E 0000F	MOVAB 1(R2), DATE_DESC+4	
				5E DD 00014	PUSHL SP	1207
00000000G	00			01 FB 00016	CALLS #1, LIB\$DATE TIME	
	09			50 E8 0001D	BLBS DATE_TIME_STATUS, 1\$	1212
				50 DD 00020	PUSHL DATE_TIME_STATUS	
00000000G	00			01 FB 00022	CALLS #1, [IB\$STOP	
	62			20 90 00029	MOVB #32, (R2)	1214
	15 A2			20 90 0002C	MOVB #32, 21(R2)	1215
	20	01		A2 91 00030	CMPB 1(R2), #32	1217
				0B 12 00034	BNEQ 2\$	
01 A2	02 A2			14 28 00036	MOVCL #20, 2(R2), 1(R2)	1220
	04 BC			15 D0 0003C	MOVL #21, @LEN	1221
				04 00040	RET	1217
	04 BC			16 D0 00041	MOVL #22, @LEN	1224
				04 00045	RET	1226

; Routine Size: 70 bytes, Routine Base: _EDT\$CODE + 0030

; 374 1227 1

```
376 1228 1 %SBTTL 'EDTSSALO_HEAP - Allocate memory'
377 1229 1
378 1230 1 GLOBAL ROUTINE EDTSSALO_HEAP (
379 1231 1     SIZE
380 1232 1     ADDRESS
381 1233 1 ) =
382 1234 1
383 1235 1 !++
384 1236 1 FUNCTIONAL DESCRIPTION:
385 1237 1
386 1238 1     Allocate memory.
387 1239 1
388 1240 1 FORMAL PARAMETERS:
389 1241 1
390 1242 1     SIZE                The number of bytes to allocate
391 1243 1
392 1244 1     ADDRESS              Place to store address of allocated space
393 1245 1
394 1246 1 IMPLICIT INPUTS:
395 1247 1
396 1248 1     NONE
397 1249 1
398 1250 1 IMPLICIT OUTPUTS:
399 1251 1
400 1252 1     NONE
401 1253 1
402 1254 1 ROUTINE VALUE:
403 1255 1
404 1256 1     1 = memory successfully allocated, 0 = out of memory.
405 1257 1
406 1258 1 SIDE EFFECTS:
407 1259 1
408 1260 1     NONE
409 1261 1
410 1262 1 !--
411 1263 1
412 1264 1 BEGIN
413 1265 1
414 1266 1 LOCAL
415 1267 1     GET_VM_STATUS;
416 1268 1
417 1269 1 IF ((.MEM_USE + ..SIZE) GTR .MEM_LIMIT) THEN RETURN (0);
418 1270 1
419 1271 1 GET_VM_STATUS = LIB$GET_VM (.SIZE, .ADDRESS);
420 1272 1
421 1273 1 IF ( NOT .GET_VM_STATUS) THEN RETURN (0);
422 1274 1
423 1275 1 MEM_USE = .MEM_USE + ..SIZE;
424 1276 1 RETURN (1);
425 1277 1 END;

! of routine EDTSSALO_HEAP
```

52 00000000' 0004 00000
EF 9E 00002

.ENTRY EDTSSALO_HEAP, Save R2
MOVAB MEM_USE, R2

: 1230

EDTSSYSVAX
V04-000

EDTSSYSVAX - VAX/VMS system specific storage
EDTSSALO_HEAP - Allocate memory

M 5
16-Sep-1984 01:52:10
14-Sep-1984 12:24:48

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]SYSVAX.B32;1

Page 13
(6)

50		62	04	BC	C1	00009	ADDL3	@SIZE, MEM_USE, R0	:	1269
	04	A2		50	D1	0000E	CMPL	R0, MEM_LIMIT	:	
				16	14	00012	BGTR	1\$:	
		7E	04	AC	7D	00014	MOVQ	SIZE, -(SP)	:	1271
00000000G		00		02	FB	00018	CALLS	#2, LIB\$GET_VM	:	
		08		50	E9	0001F	BLBC	GET_VM_STATUS, 1\$:	1273
		62	04	BC	C0	00022	ADDL2	@SIZE, -MEM_USE	:	1275
		50		01	D0	00026	MOVL	#1, R0	:	1276
					04	00029	RET		:	
				50	D4	0002A	CLRL	R0	:	1277
					04	0002C	RET		:	

; Routine Size: 45 bytes, Routine Base: _EDT\$CODE + 0076

; 426 1278 1

```
428 1279 1 %SBTTL 'EDTSSDEA_HEAP - Deallocate memory'
429 1280 1
430 1281 1 GLOBAL ROUTINE EDTSSDEA_HEAP (
431 1282 1     SIZE,
432 1283 1     ADDRESS
433 1284 1 ) : NOVALUE =
434 1285 1
435 1286 1 ++
436 1287 1 FUNCTIONAL DESCRIPTION:
437 1288 1
438 1289 1     Deallocate memory.
439 1290 1
440 1291 1 FORMAL PARAMETERS:
441 1292 1
442 1293 1     SIZE                The number of bytes to deallocate
443 1294 1
444 1295 1     ADDRESS              Place to store address of deallocated space
445 1296 1
446 1297 1 IMPLICIT INPUTS:
447 1298 1
448 1299 1     NONE
449 1300 1
450 1301 1 IMPLICIT OUTPUTS:
451 1302 1
452 1303 1     NONE
453 1304 1
454 1305 1 ROUTINE VALUE:
455 1306 1
456 1307 1     NONE
457 1308 1
458 1309 1 SIDE EFFECTS:
459 1310 1
460 1311 1     Signals on error.
461 1312 1
462 1313 1 --
463 1314 1
464 1315 2 BEGIN
465 1316 2
466 1317 2 LOCAL
467 1318 2     FREE_VM_STATUS;
468 1319 2
469 1320 2     FREE_VM_STATUS = LIB$FREE_VM (.SIZE, .ADDRESS);
470 1321 2
471 1322 2     IF ( NOT .FREE_VM_STATUS) THEN SIGNAL_STOP (.FREE_VM_STATUS);
472 1323 2
473 1324 2     MEM_USE = .MEM_USE - .SIZE;
474 1325 2     ASSERT (.MEM_USE GEQ 0);
475 1326 1 END;
```

! of routine EDTSSDEA_HEAP

```
00000000G 7E 04 AC 7D 00002
00 02 FB 00006
09 50 EB 0000D
```

```
.ENTRY EDTSSDEA_HEAP, Save nothing
MOVQ SIZE, -(SP)
CALLS #2, LIB$FREE_VM
BLBS FREE_VM_STATUS, 1$
```

```
: 1281
: 1320
: 1322
```

EDTSSYSVAX
V04-000

EDTSSYSVAX - VAX/VMS system specific storage
EDTSSDEA_HEAP - Deallocate memory

16-Sep-1984 01:52:10
14-Sep-1984 12:24:48

VAX-11 BLISS-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]SYSVAX.B32;1

Page 15
(7)

00000000G	00	50	DD	00010	PUSHL	FREE_VM_STATUS	
00000000	EF	01	FB	00012	CALLS	#1, CIB\$STOP	
		04	BC	00019	SUBL2	@SIZE, MEM_USE	1324
		07	18	00021	BGEQ	28	1325
00000000G	00	00	FB	00023	CALLS	#0, EDT\$\$INTER_ERR	
			04	0002A	RET		1326

: Routine Size: 43 bytes, Routine Base: _EDT\$CODE + 00A3

: 476 1327 1


```
478 1328 1 XSBTTL 'EDTSSDEA_ALLHEAP - Deallocate all memory'
479 1329 1
480 1330 1 GLOBAL ROUTINE EDTSSDEA_ALLHEAP          ! Deallocate all memory
481 1331 1 : NOVALUE =
482 1332 1
483 1333 1
484 1334 1 ++
485 1335 1 FUNCTIONAL DESCRIPTION:
486 1336 1     Deallocate all memory allocated by calls to LIB$GET_VM .
487 1337 1
488 1338 1 FORMAL PARAMETERS:
489 1339 1
490 1340 1     NONE
491 1341 1
492 1342 1 IMPLICIT INPUTS:
493 1343 1
494 1344 1     EDTSSA_FST_AVLN
495 1345 1     EDTSSA_FST_SCRPTR
496 1346 1     EDTSSA_BUF_LST
497 1347 1     EDTSSA_TRN_TBL
498 1348 1     EDTSSA_US_ENT
499 1349 1     EDTSSA_US_TXT
500 1350 1
501 1351 1 IMPLICIT OUTPUTS:
502 1352 1
503 1353 1     EDTSSA_FST_AVLN
504 1354 1     EDTSSA_FST_SCRPTR
505 1355 1     EDTSSA_BUF_LST
506 1356 1
507 1357 1 ROUTINE VALUE:
508 1358 1
509 1359 1     NONE
510 1360 1
511 1361 1 SIDE EFFECTS:
512 1362 1
513 1363 1     Signals on error.
514 1364 1
515 1365 1 --
516 1366 1
517 1367 1 BEGIN
518 1368 1
519 1369 1 EXTERNAL ROUTINE
520 1370 1     STR$FREE1 DX,
521 1371 1     EDTSSCAN_RDEF;          ! Cancel a key definition
522 1372 1
523 1373 1 EXTERNAL
524 1374 1     EDTSSA_FST_AVLN,
525 1375 1     EDTSSA_FST_SCRPTR,
526 1376 1     EDTSSA_BUF_LST,
527 1377 1     EDTSSA_TRN_TBL : VECTOR,
528 1378 1     EDTSSA_US_ENT : VECTOR,
529 1379 1     EDTSSA_US_TXT : VECTOR;
530 1380 1
531 1381 1 LOCAL
532 1382 1     NEW_PTR : REF SCREEN_LINE,
533 1383 1     NEW_BUF : REF TBCB_BLOCK,
534 1384 1     LEN;
```

```
535 1385 GET_VM_STATUS;
536 1386
537 1387
538 1388
539 1389
540 1390 NEW_BUF = .EDTSSA_BUF_LST;
541 1391
542 1392 WHILE (.NEW_BUF NEQA 0) DO
543 1393 BEGIN
544 1394     LEN = .NEW_BUF [TBCB_NAME_LEN] + TBCB_SIZE;
545 1395     EDTSSA_BUF_LST = .NEW_BUF [TBCB_NEXT_BUF];
546 1396     EDTSSDEA_HEAP (LEN, NEW_BUF);
547 1397     NEW_BUF = .EDTSSA_BUF_LST;
548 1398 END;
549 1399
550 1400
551 1401
552 1402
553 1403 NEW_PTR = .EDTSSA_FST_SCRPTR;
554 1404
555 1405 WHILE (.NEW_PTR NEQA 0) DO
556 1406 BEGIN
557 1407     EDTSSA_FST_SCRPTR = .NEW_PTR [SCR_NXT_LINE];
558 1408     EDTSSDEA_HEAP (%REF (SCR_SIZE), NEW_PTR);
559 1409     NEW_PTR = .EDTSSA_FST_SCRPTR;
560 1410 END;
561 1411
562 1412 NEW_PTR = .EDTSSA_FST_AVLN;
563 1413
564 1414 WHILE (.NEW_PTR NEQA 0) DO
565 1415 BEGIN
566 1416     EDTSSA_FST_AVLN = .NEW_PTR [SCR_NXT_LINE];
567 1417     EDTSSDEA_HEAP (%REF (SCR_SIZE), NEW_PTR);
568 1418     NEW_PTR = .EDTSSA_FST_AVLN;
569 1419 END;
570 1420
571 1421
572 1422
573 1423
574 1424
575 1425 INCR ENT_NUM FROM 0 TO 3 DO
576 1426 BEGIN
577 1427     LEN = CH$RCHAR (.EDTSSA_US_ENT [.ENT_NUM]);
578 1428     EDTSSDEA_HEAP (%REF (.LEN + 1), EDTSSA_US_ENT [.ENT_NUM]);
579 1429 END;
580 1430
581 1431 INCR TEXT_NUM FROM 0 TO 1 DO
582 1432 BEGIN
583 1433     LEN = CH$RCHAR (.EDTSSA_US_TXT [.TEXT_NUM]);
584 1434     EDTSSDEA_HEAP (%REF (.LEN + 1), EDTSSA_US_TXT [.TEXT_NUM]);
585 1435 END;
586 1436
587 1437
588 1438
589 1439
590 1440
591 1441 INCR TBL_PTR FROM 0 TO K_KPAD_HASHSZ - 1 DO
```

```

592      1442      BEGIN
593      1443
594      1444      WHILE (.EDTSSA_TRN_TBL [.TBL_PTR] NEQA 0) DO
595      1445          BEGIN
596      1446              LOCAL
597      1447                  KEY_PTR : REF BLOCK [, BYTE] FIELD (KEY_DEF_FIELD);
598      1448
599      1449              KEY_PTR = .EDTSSA_TRN_TBL [.TBL_PTR];
600      1450              EDTSSCAN_KDEF (.KEY_PTR [KEY_DEF_KEY]);
601      1451          END;
602      1452      END;
603      1453
604      1454      END;
605      1455
606      1456      END;

```

PC	Op	OpC	OpD	OpE	OpF	OpG	OpH	OpI	OpJ	OpK	OpL	OpM	OpN	OpO	OpP	OpQ	OpR	OpS	OpT	OpU	OpV	OpW	OpX	OpY	OpZ	OpAA	OpAB	OpAC	OpAD	OpAE	OpAF	OpAG	OpAH	OpAI	OpAJ	OpAK	OpAL	OpAM	OpAN	OpAO	OpAP	OpAQ	OpAR	OpAS	OpAT	OpAU	OpAV	OpAW	OpAX	OpAY	OpAZ	OpBA	OpBB	OpBC	OpBD	OpBE	OpBF	OpBG	OpBH	OpBI	OpBJ	OpBK	OpBL	OpBM	OpBN	OpBO	OpBP	OpBQ	OpBR	OpBS	OpBT	OpBU	OpBV	OpBW	OpBX	OpBY	OpBZ	OpCA	OpCB	OpCC	OpCD	OpCE	OpCF	OpCG	OpCH	OpCI	OpCJ	OpCK	OpCL	OpCM	OpCN	OpCO	OpCP	OpCQ	OpCR	OpCS	OpCT	OpCU	OpCV	OpCW	OpCX	OpCY	OpCZ	OpDA	OpDB	OpDC	OpDD	OpDE	OpDF	OpDG	OpDH	OpDI	OpDJ	OpDK	OpDL	OpDM	OpDN	OpDO	OpDP	OpDQ	OpDR	OpDS	OpDT	OpDU	OpDV	OpDW	OpDX	OpDY	OpDZ	OpEA	OpEB	OpEC	OpED	OpEE	OpEF	OpEG	OpEH	OpEI	OpEJ	OpEK	OpEL	OpEM	OpEN	OpEO	OpEP	OpEQ	OpER	OpES	OpET	OpEU	OpEV	OpEW	OpEX	OpEY	OpEZ	OpFA	OpFB	OpFC	OpFD	OpFE	OpFF	OpFG	OpFH	OpFI	OpFJ	OpFK	OpFL	OpFM	OpFN	OpFO	OpFP	OpFQ	OpFR	OpFS	OpFT	OpFU	OpFV	OpFW	OpFX	OpFY	OpFZ	OpGA	OpGB	OpGC	OpGD	OpGE	OpGF	OpGG	OpGH	OpGI	OpGJ	OpGK	OpGL	OpGM	OpGN	OpGO	OpGP	OpGQ	OpGR	OpGS	OpGT	OpGU	OpGV	OpGW	OpGX	OpGY	OpGZ	OpHA	OpHB	OpHC	OpHD	OpHE	OpHF	OpHG	OpHH	OpHI	OpHJ	OpHK	OpHL	OpHM	OpHN	OpHO	OpHP	OpHQ	OpHR	OpHS	OpHT	OpHU	OpHV	OpHW	OpHX	OpHY	OpHZ	OpIA	OpIB	OpIC	OpID	OpIE	OpIF	OpIG	OpIH	OpII	OpIJ	OpIK	OpIL	OpIM	OpIN	OpIO	OpIP	OpIQ	OpIR	OpIS	OpIT	OpIU	OpIV	OpIW	OpIX	OpIY	OpIZ	OpJA	OpJB	OpJC	OpJD	OpJE	OpJF	OpJG	OpJH	OpJI	OpJJ	OpJK	OpJL	OpJM	OpJN	OpJO	OpJP	OpJQ	OpJR	OpJS	OpJT	OpJU	OpJV	OpJW	OpJX	OpJY	OpJZ	OpKA	OpKB	OpKC	OpKD	OpKE	OpKF	OpKG	OpKH	OpKI	OpKJ	OpKK	OpKL	OpKM	OpKN	OpKO	OpKP	OpKQ	OpKR	OpKS	OpKT	OpKU	OpKV	OpKW	OpKX	OpKY	OpKZ	OpLA	OpLB	OpLC	OpLD	OpLE	OpLF	OpLG	OpLH	OpLI	OpLJ	OpLK	OpLL	OpLM	OpLN	OpLO	OpLP	OpLQ	OpLR	OpLS	OpLT	OpLU	OpLV	OpLW	OpLX	OpLY	OpLZ	OpMA	OpMB	OpMC	OpMD	OpME	OpMF	OpMG	OpMH	OpMI	OpMJ	OpMK	OpML	OpMM	OpMN	OpMO	OpMP	OpMQ	OpMR	OpMS	OpMT	OpMU	OpMV	OpMW	OpMX	OpMY	OpMZ	OpNA	OpNB	OpNC	OpND	OpNE	OpNF	OpNG	OpNH	OpNI	OpNJ	OpNK	OpNL	OpNM	OpNN	OpNO	OpNP	OpNQ	OpNR	OpNS	OpNT	OpNU	OpNV	OpNW	OpNX	OpNY	OpNZ	OpOA	OpOB	OpOC	OpOD	OpOE	OpOF	OpOG	OpOH	OpOI	OpOJ	OpOK	OpOL	OpOM	OpON	OpOO	OpOP	OpOQ	OpOR	OpOS	OpOT	OpOU	OpOV	OpOW	OpOX	OpOY	OpOZ	OpPA	OpPB	OpPC	OpPD	OpPE	OpPF	OpPG	OpPH	OpPI	OpPJ	OpPK	OpPL	OpPM	OpPN	OpPO	OpPP	OpPQ	OpPR	OpPS	OpPT	OpPU	OpPV	OpPW	OpPX	OpPY	OpPZ	OpQA	OpQB	OpQC	OpQD	OpQE	OpQF	OpQG	OpQH	OpQI	OpQJ	OpQK	OpQL	OpQM	OpQN	OpQO	OpQP	OpQQ	OpQR	OpQS	OpQT	OpQU	OpQV	OpQW	OpQX	OpQY	OpQZ	OpRA	OpRB	OpRC	OpRD	OpRE	OpRF	OpRG	OpRH	OpRI	OpRJ	OpRK	OpRL	OpRM	OpRN	OpRO	OpRP	OpRQ	OpRR	OpRS	OpRT	OpRU	OpRV	OpRW	OpRX	OpRY	OpRZ	OpSA	OpSB	OpSC	OpSD	OpSE	OpSF	OpSG	OpSH	OpSI	OpSJ
----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------

			E3	11	00078	BRB	3\$	1418
			52	D4	0007A	CLRL	ENT_NUM	1425
			42	DE	0007C	MOVAL	EDTSSA_US ENT[ENT_NUM], R0	1427
	08	AE	00	80	9A	MOVZBL	20(R0), LEN	
			50	DD	00089	PUSHL	R0	1428
04	AE	0C	AE	01	C1	ADDL3	#1, LEN, 4(SP)	
			04	AE	9F	PUSHAB	4(SP)	
		63		02	FB	CALLS	#2, EDTSSDEA_HEAP	
E1		52		03	F3	AOBLEQ	#3, ENT_NUM, -5\$	1425
			52	D4	0009B	CLRL	TEXT_NUM	1431
			42	DE	0009D	MOVAL	EDTSSA_US TXT[TEXT_NUM], R0	1433
	08	AE	00	80	9A	MOVZBL	20(R0), LEN	
			50	DD	000AA	PUSHL	R0	1434
04	AE	0C	AE	01	C1	ADDL3	#1, LEN, 4(SP)	
			04	AE	9F	PUSHAB	4(SP)	
		63		02	FB	CALLS	#2, EDTSSDEA_HEAP	
E1		52		01	F3	AOBLEQ	#1, TEXT_NUM, 6\$	1431
			52	D4	000BC	CLRL	TBL_PTR	1441
			42	D0	000BE	MOVL	EDTSSA_TRN_TBL[TBL_PTR], R0	1444
			0D	13	000C6	BEQL	8\$	
		7E	04	A0	3C	MOVZWL	4(KEY_PTR), -(SP)	1451
		00		01	FB	CALLS	#1, EDTSSCAN_KDEF	
			E9	11	000D3	BRB	7\$	1444
E1		52	000000C6	8F	F3	AOBLEQ	#198, TBL_PTR, 7\$	1441
				04	000DD	RET		1456

; Routine Size: 222 bytes, Routine Base: _EDT\$CODE + 00CE

; 607 1457 1

```

609 1458 1 %SBTTL 'WORKAST - take a timer AST for the WORKING message'
610 1459 1 ROUTINE WORKAST ! Take a timer AST for the WORKING message
611 1460 1 : NOVALUE =
612 1461 1
613 1462 1
614 1463 1 ++
615 1464 1 FUNCTIONAL DESCRIPTION:
616 1465 1 Take a timer AST for the WORKING message.
617 1466 1
618 1467 1 FORMAL PARAMETERS:
619 1468 1
620 1469 1 NONE
621 1470 1
622 1471 1 IMPLICIT INPUTS:
623 1472 1
624 1473 1 WORK_MESSAGE_RUNNING
625 1474 1
626 1475 1 IMPLICIT OUTPUTS:
627 1476 1
628 1477 1 EDT$$G_SECOND
629 1478 1
630 1479 1 ROUTINE VALUE:
631 1480 1
632 1481 1 NONE
633 1482 1
634 1483 1 SIZE EFFECTS:
635 1484 1
636 1485 1 Arranges to print the WORKING message on the screen.
637 1486 1
638 1487 1 --
639 1488 1
640 1489 2 BEGIN
641 1490 2
642 1491 2 IF .WORK_MESSAGE_RUNNING
643 1492 2 THEN
644 1493 2 BEGIN
645 1494 2 EDT$$G_SECOND = 1;
646 1495 2 $SETIMR (DAYTIM = DEL_TIME, ASTADR = WORKAST, REQIDT = EDT$$G_WORKCOUNT);
647 1496 2 END;
648 1497 2
649 1498 1 END; ! of routine WORKAST
```

```

                                .EXTRN SYS$SETIMR
                                0000 00000 WORKAST: .WORD Save nothing
                                0000 00002          BLBC WORK MESSAGE RUNNING, 1$
00000000G 1F 00000000' EF E9 00002          MOVL #1, EDT$$G_SECOND
                                00 00 00009          PUSHAB EDT$$G_WORKCOUNT
                                00 9F 00010          PUSHAB WORKAST
                                AF 9F 00016          PUSHAB DEL TIME
                                EF 9F 00019          CLRL -(SP)
                                7E D4 0001F          CALLS #4, SYS$SETIMR
00000000G 00 04 FB 00021          RET
                                04 00028 1$:
```

; Routine Size: 41 bytes, Routine Base: _EDT\$CODE + 01AC

EDTSSYSVAX
V04-000

EDTSSYSVAX - VAX/VMS system specific storage
WORKAST - take a timer AST for the WORKING mess

H 6
16-Sep-1984 01:52:10
14-Sep-1984 12:24:48

VAX-11 Bliss-32 V4 0-742
DISK\$VMSMASTER:[EDT.SRC]SYSVAX.B32;1 Page 21 (9)

```

651 1499 1 %SBTTL 'EDTSSSTART_WKINGMSG - initiate the 'working' timer'
652 1500 1
653 1501 1 GLOBAL ROUTINE EDTSSSTART_WKINGMSG ! Initiate the 'working' timer
654 1502 1 : NOVALUE =
655 1503 1
656 1504 1 ++
657 1505 1 FUNCTIONAL DESCRIPTION:
658 1506 1
659 1507 1 Start the timer which will cause the 'working' message
660 1508 1 to print occasionally until it is cancelled.
661 1509 1
662 1510 1 FORMAL PARAMETERS:
663 1511 1
664 1512 1 NONE
665 1513 1
666 1514 1 IMPLICIT INPUTS:
667 1515 1
668 1516 1 DEL TIME
669 1517 1 WORKAST
670 1518 1 WORK_MESSAGE_RUNNING
671 1519 1
672 1520 1 IMPLICIT OUTPUTS:
673 1521 1
674 1522 1 EDTSSG_WORKCOUNT
675 1523 1 WORKING_EFN
676 1524 1 WORK_MESSAGE_RUNNING
677 1525 1
678 1526 1 ROUTINE VALUE:
679 1527 1
680 1528 1 NONE
681 1529 1
682 1530 1 SIDE EFFECTS:
683 1531 1
684 1532 1 Allocates an event flag.
685 1533 1 Signals any errors.
686 1534 1
687 1535 1 --
688 1536 1
689 1537 2 BEGIN
690 1538 2
691 1539 2 LOCAL
692 1540 2 GETEF_STATUS,
693 1541 2 SETIMR_STATUS;
694 1542 2
695 1543 2 ++
696 1544 2 If the 'working' message is already running, don't start it again.
697 1545 2
698 1546 2
699 1547 2 IF .WORK_MESSAGE_RUNNING THEN RETURN;
700 1548 2
701 1549 2 GETEF_STATUS = LIB$GET_EF (WORKING_EFN);
702 1550 2
703 1551 2 IF ( NOT .GETEF_STATUS) THEN SIGNAL_STOP (.GETEF_STATUS);
704 1552 2
705 P 1553 2 SETIMR STATUS = $SETIMR (EFN = .WORKING_EFN, DAYTIM = DEL_TIME, ASTADR = WORKAST,
706 1554 2 REQIDT = EDTSSG_WORKCOUNT);
707 1555 2
```


EDTSSYSVAX
V04-000

EDTSSYSVAX - VAX/VMS system specific storage.
EDTSSSTART_WKINGMSG - initiate the "working" t

J 6
16-Sep-1984 01:52:10
14-Sep-1984 12:24:48

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]SYSVAX.B32;1 Page 23
(10)

```
: 708      1556 2      IF ( NOT .SETIMR_STATUS) THEN SIGNAL_STOP (.SETIMR_STATUS);
: 709      1557 2
: 710      1558 2      EDTSSG WORKCOUNT = 0;
: 711      1559 2      WORK_MESSAGE_RUNNING = 1;
: 712      1560 1      END;
```

! of routine EDTSSSTART_WKINGMSG

			001C 00000	.ENTRY EDTSSSTART_WKINGMSG, Save R2,R3,R4	1501
	54 00000000G	00	9E 00002	MOVAB EDTSSG WORKCOUNT, R4	
	53 00000000G	00	9E 00009	MOVAB LIB\$STOP, R3	
	52 00000000G	EF	9E 00010	MOVAB WORK_MESSAGE_RUNNING, R2	
	31	62	EB 00017	BLBS WORK_MESSAGE_RUNNING, 3\$	1547
		A2	9F 0001A	PUSHAB WORKING_EFN	1549
00000000G	00	01	FB 0001D	CALLS #1, LIB\$GET_EF	
	05	50	EB 00024	BLBS GETEF_STATUS, 1\$	1551
		50	DD 00027	PUSHL GETEF_STATUS	
	63	01	FB 00029	CALLS #1, LIB\$STOP	
		54	DD 0002C 1\$:	PUSHL R4	1554
		A6	9F 0002E	PUSHAB WORKAST	
		F4	9F 00031	PUSHAB DEL TIME	
		A2	DD 00034	PUSHL WORKING_EFN	
00000000G	00	04	FB 00037	CALLS #4, SYS\$SETIMR	
	05	50	EB 0003E	BLBS SETIMR_STATUS, 2\$	1556
		50	DD 00041	PUSHL SETIMR_STATUS	
	63	01	FB 00043	CALLS #1, LIB\$STOP	
		64	D4 00046 2\$:	CLRL EDTSSG WORKCOUNT	1558
	62	01	D0 00048	MOVL #1, WORK_MESSAGE_RUNNING	1559
		04	0004B 3\$:	RET	1560

; Routine Size: 76 bytes, Routine Base: _EDT\$CODE + 01D5

: 713 1561 1

```

715 1562 1 %SBTTL 'EDT$STOP_WKINGMSG - cancel the 'working' timer'
716 1563 1
717 1564 1 GLOBAL ROUTINE EDT$STOP_WKINGMSG ! Cancel the 'working' timer
718 1565 1 : NOVALUE =
719 1566 1
720 1567 1 ++
721 1568 1 FUNCTIONAL DESCRIPTION:
722 1569 1
723 1570 1 Cancel the 'working' timer. The 'working' message will not print
724 1571 1 until it is initiated again. Also, erase the working message.
725 1572 1
726 1573 1 FORMAL PARAMETERS:
727 1574 1
728 1575 1 NONE
729 1576 1
730 1577 1 IMPLICIT INPUTS:
731 1578 1
732 1579 1 WORKING_EFN
733 1580 1 EDT$G_WORKCOUNT
734 1581 1 WORK_MESSAGE_RUNNING
735 1582 1 EDT$G_MESSAGE_LINE
736 1583 1
737 1584 1 IMPLICIT OUTPUTS:
738 1585 1
739 1586 1 WORK_MESSAGE_RUNNING
740 1587 1
741 1588 1 ROUTINE VALUE:
742 1589 1
743 1590 1 NONE
744 1591 1
745 1592 1 SIDE EFFECTS:
746 1593 1
747 1594 1 Deallocates an event flag.
748 1595 1 Repositions the cursor to beginning of message line
749 1596 1
750 1597 1 --
751 1598 1
752 1599 2 BEGIN
753 1600 2
754 1601 2 LOCAL
755 1602 2 FORMAT_ROUTINE,
756 1603 2 FREEEF_STATUS,
757 1604 2 CANTIM_STATUS;
758 1605 2
759 1606 2 ++
760 1607 2 If the 'working' message is not running, do nothing.
761 1608 2
762 1609 2
763 1610 2 IF ( NOT .WORK_MESSAGE_RUNNING) THEN RETURN;
764 1611 2
765 1612 2 WORK_MESSAGE_RUNNING = 0;
766 1613 2 CANTIM_STATUS = $CANTIM (REQIDT = EDT$G_WORKCOUNT);
767 1614 2
768 1615 2 IF ( NOT .CANTIM_STATUS) THEN SIGNAL_STOP (.CANTIM_STATUS);
769 1616 2
770 1617 2 FREEEF_STATUS = LIB$FREE_EF (WORKING_EFN);
771 1618 2
```

```

772 1619 2      IF ( NOT .FREEEF_STATUS) THEN SIGNAL_STOP (.FREEEF_STATUS);
773 1620
774 1621
775 1622  !+ Erase the working message when it is stopped if not already done
776 1623  !-
777 1624      FORMAT_ROUTINE = .EDTSSA_FMT_WRRUT;
778 1625      EDTSSA_FMT_WRRUT = EDTSSI_WSTR;
779 1626
780 1627      IF (.EDTSSG_WORKCOUNT)
781 1628      THEN
782 1629          BEGIN
783 1630              EDTSSC_POSCSIF (.EDTSSG_MESSAGE_LINE + 1, 0);
784 1631              EDTSSC_ERATOEOI ();
785 1632              EDTSSOUT_FMTBUF ();
786 1633          END;
787 1634
788 1635  !+ If "working" was printed then reposition the cursor to the left-most
789 1636  !- position of the prompt.
790 1637
791 1638
792 1639
793 1640      IF (.EDTSSG_WORKCOUNT NEQ 0)
794 1641      THEN
795 1642          BEGIN
796 1643              EDTSSC_POSCSIF (.EDTSSG_MESSAGE_LINE + 1, 0);
797 1644              EDTSSOUT_FMTBUF ();
798 1645          END;
799 1646
800 1647      EDTSSA_FMT_WRRUT = .FORMAT_ROUTINE;
801 1648      EDTSSG_SECOND = 0;
802 1649  END;
```

! of routine EDTSSSTOP_WKINGMSG

				.EXTRN	SYSSCANTIM	
				.ENTRY	EDTSSSTOP_WKINGMSG, Save R2,R3,R4,R5,R6,R7,-;	1564
					R8,R9	
				MOVAB	EDTSSOUT_FMTBUF, R9	
				MOVAB	EDTSSC_POSCSIF, R8	
				MOVAB	EDTSSG_MESSAGE_LINE, R7	
				MOVAB	LIBSTOP, R6	
				MOVAB	WORK_MESSAGE_RUNNING, R5	
				MOVAB	EDTSSA_FMT_WRRUT, R4	
				MOVAB	EDTSSG_WORKCOUNT, R3	
				BLBC	WORK_MESSAGE_RUNNING, 58	1610
				CLRL	WORK_MESSAGE_RUNNING	1612
				CLRL	-(SP)	1613
				PUSHL	R3	
				CALLS	#2, SYSSCANTIM	
				BLBS	CANTIM_STATUS, 18	1615
				PUSHL	CANTIM_STATUS	
				CALLS	#1, LIBSTOP	
				PUSHAB	WORKING_EFN	1617
				CALLS	#1, LIB\$FREE_EF	
				BLBS	FREEEF_STATUS, 28	1619
				PUSHL	FREEEF_STATUS	

				03FC 00000	
59	00000000G	00	9E 00002		
58	00000000G	00	9E 00009		
57	00000000G	00	9E 00010		
56	00000000G	00	9E 00017		
55	00000000'	EF	9E 0001E		
54	00000000G	00	9E 00025		
53	00000000G	00	9E 0002C		
60		65	E9 00033		
		65	D4 00036		
		7E	D4 00038		
		53	DD 0003A		
00000000G	00	02	FB 0003C		
	05	50	E8 00043		
		50	DD 00046		
	66	01	FB 00048		
00000000G	00	FC	A5 9F 0004B	18:	
		01	FB 0004E		
	05	50	E8 00055		
		50	DD 00058		

EDTSSYSVAX
V04-000

EDTSSYSVAX - VAX/VMS system specific storage
EDTSSSTOP_WKINGMSG - cancel the "working" time

M 6

16-Sep-1984 01:52:10

14-Sep-1984 12:24:48

VAX-11 Bliss-32 V4.0-742

DISK\$VMSMASTER:[EDT.SRC]SYSVAX.B32;1

Page 26

(11)

	66	01	FB	0005A	CALLS	#1, LIB\$STOP	
	52	64	D0	0005D	MOVL	EDTSSA_FMT_WRRUT, FORMAT ROUTINE	1624
	64	00	9E	00060	MOVAB	EDTSSIT_WRRUT, EDTSSA_FMT_WRRUT	1625
	13	63	E9	00067	BLBC	EDTSSG_WORKCOUNT, 3\$	1627
		7E	D4	0006A	CLRL	-(SP)	1630
7E	67	01	C1	0006C	ADDL3	#1, EDTSSG_MESSAGE_LINE, -(SP)	
	68	02	FB	00070	CALLS	#2, EDTSSC_POSCSIF	
00000000G	00	00	FB	00073	CALLS	#0, EDTSSC_ERATOEOI	1631
	69	00	FB	0007A	CALLS	#0, EDTSSOUT_FMTBUF	1632
		63	D5	0007D	TSTL	EDTSSG_WORKCOUNT	1640
		0C	13	0007F	BEQL	4\$	
		7E	D4	00081	CLRL	-(SP)	1643
7E	67	01	C1	00083	ADDL3	#1, EDTSSG_MESSAGE_LINE, -(SP)	
	68	02	FB	00087	CALLS	#2, EDTSSC_POSCSIF	
	69	00	FB	0008A	CALLS	#0, EDTSSOUT_FMTBUF	1644
	64	52	D0	0008D	MOVL	FORMAT ROUTINE, EDTSSA_FMT_WRRUT	1647
00000000G	00	D4	00090	CLRL	EDTSSG_SECOND		1648
		04	00096	NET			1649

; Routine Size: 151 bytes, Routine Base: _EDT\$CODE + 0221

; 803 1650 1


```

: 805 1651 1 %SBTTL 'EDT$$MSG_TOSTR - print a system message'
: 806 1652 1
: 807 1653 1 GLOBAL ROUTINE EDT$$MSG_TOSTR (          ! Print a system message
: 808 1654 1     MESS_NUM                          ! message number
: 809 1655 1     ) : NOVALUE =
: 810 1656 1
: 811 1657 1 ++
: 812 1658 1 FUNCTIONAL DESCRIPTION:
: 813 1659 1
: 814 1660 1     Print a system message, given its message number.
: 815 1661 1
: 816 1662 1 FORMAL PARAMETERS:
: 817 1663 1
: 818 1664 1     MESS_NUM          The number of the message to print
: 819 1665 1
: 820 1666 1 IMPLICIT INPUTS:
: 821 1667 1
: 822 1668 1     NONE
: 823 1669 1
: 824 1670 1 IMPLICIT OUTPUTS:
: 825 1671 1
: 826 1672 1     NONE
: 827 1673 1
: 828 1674 1 ROUTINE VALUE:
: 829 1675 1
: 830 1676 1     NONE
: 831 1677 1
: 832 1678 1 SIDE EFFECTS:
: 833 1679 1
: 834 1680 1     Prints a message on the terminal.
: 835 1681 1
: 836 1682 1 --
: 837 1683 1
: 838 1684 2 BEGIN
: 839 1685 2
: 840 1686 2 LOCAL
: 841 1687 2     MSGBUF : BLOCK [CH$ALLOCATION (80)],
: 842 1688 2     MSGDESC : VECTOR [2],
: 843 1689 2     MSGLEN;
: 844 1690 2
: 845 1691 2     MSGDESC [0] = 80;
: 846 1692 2     MSGDESC [1] = MSGBUF;
: 847 1693 2     $GETMSG (MSGID = .MESS_NUM, MSGLEN = MSGLEN, BUFADR = MSGDESC, FLAGS = 1);
: 848 1694 2     EDT$$FMT_STR (MSGBUF, .MSGLEN<0, 16>);
: 849 1695 1 END;

```

! of routine EDT\$\$MSG_TOSTR

.EXTRN SYS\$GETMSG

```

          0000 00000
04      SE    A4    AE    9E 00002
08      AE    50    8F    9A 00006
        AE    0C    AE    9E 0000B
        7E    01    7D    00010
        OC    AE    9F    00013
        OC    AE    9F    00016

```

```

.ENTRY EDT$$MSG_TOSTR, Save nothing
MOVAB  -92(SP), SP
MOVZBL #80, MSGDESC
MOVAB  MSGBUF, MSGDESC+4
MOVQ   #1, -(SP)
PUSHAB MSGDESC
PUSHAB MSGLEN

```

```

: 1653
: 1691
: 1692
: 1693
:

```

000000000G	00	04	AC	DD	00019
	7E		05	FB	0001C
			6E	3C	00023
		10	AE	9F	00026
000000000G	00		02	FB	00029
				04	00030

```

PUSHL      MESS_NUM
CALLS      #5, SYS$GETMSG
MOVZWL     MSGLEN, -(SP)
PUSHAB     MSGBUF
CALLS      #2, EDT$$FMT_STR
RET

```

1694
1695

```
; Routine Size: 49 bytes,    Routine Base: _EDT$CODE + 02BB
```

```

: 850      1696 1
: 851      1697 1 !<BLF/PAGE>

```

EDT\$SYSVAX
V04-000

EDT\$SYSVAX - VAX/VMS system specific storage
EDT\$\$MSG_TOSTR - print a system message

C 7
16-Sep-1984 01:52:10
14-Sep-1984 12:24:48

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]SYSVAX.B32;1 Page 29 (13)

: 853 1698 1 END
: 854 1699 1
: 855 1700 0 ELUDOM

! of module EDT\$SYSVAX

.EXTRN LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
EDT\$DATA	644	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, PIC, ALIGN(2)
EDT\$CODE	745	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	14	0	581	00:02.6
\$255\$DUA28:[EDT.SRC]EDT.L32;1	377	37	9	40	00:00.8
\$255\$DUA28:[EDT.SRC]PSECTS.L32;1	2	1	50	7	00:00.1
\$255\$DUA28:[EDT.SRC]KEYPADDEF.L32;1	34	6	17	7	00:00.2

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACEBACK/LIS=LIS\$:SYSVAX/OBJ=OBJ\$:SYSVAX MSRC\$:SYSVAX.B32/UPDATE=(ENH\$:SYSVAX)

: Size: 745 code + 644 data bytes
: Run Time: 00:43.4
: Elapsed Time: 00:58.6
: Lines/CPU Min: 2351
: Lexemes/CPU-Min: 8231
: Memory Used: 153 pages
: Compilation Complete

0140

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY